PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellant:	Dally et al.	Examiner:	Meonske, T.
Application No.:	09/871,301	Group Art Unit:	2183
Filed:	May 30, 2001	Docket No.:	STFD.076CIP (S97-246CIP)
Title:	System And Method For Performing Efficient Conditional Vector Operations For Data Parallel Architectures Involving Both Input And Conditional Vector Values		

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence and the papers, as described hereinabove, are being deposited in the United States Postal Service, as first class mail, in an envelope addressed to: Board of Patent Appeals and Interferences, United States Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450, on September 8, 2005.

By: Kelly S. Waltigney
Kelly S. Waltigney

AMENDED APPEAL BRIEF

Board of Patent Appeals and Interferences
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

Customer No. 40581

Sir:

In response to the Notification of Non-Compliant Appeal Brief dated August 9, 2005, this is an Amended Appeal Brief submitted pursuant to 37 C.F.R. § 41.37(d) for the above-referenced patent application with the requested changes being made to the summary (section V).

Please charge Deposit Account No. 50-0996 (STFD.076CIP) for any fees related to this brief in support of appeal as indicated in 37 C.F.R. § 41.20(b)(2).

I. Real Party in Interest

The real parties in interest are the assignees, The Board of Trustees of the Leland Stanford Junior University and The Massachusetts Institute of Technology.

II. Related Appeals and Interferences

Appellant is unaware of any related appeals, interferences or judicial proceedings that would have a bearing on the Board's decision in the instant appeal.

III. Status of Claims

Claims 10-17, 20, 21, 24 and 26 are presented for appeal and each of the appealed claims, 10-17, 20, 21, 24 and 26, is rejected. Claims 1-9, 22, 23 and 25 have been canceled and claims 18-19 have been allowed. The pending claims under appeal, as presently amended, may be found in the attached Appendix of Appealed Claims.

IV. Status of Amendments

In the Office Action Response After Final filed on December 30, 2004, claims 1-9 were canceled. The Advisory Action dated February 11, 2005 indicates that this amendment has been entered (canceling claims 1-9).

V. Summary of Invention

The independent claims involved in the appeal (claims 10, 12-17, 20, 21, 24 and 26) are generally directed to providing a high level of productive processor cycle usage by providing adequate, yet efficient, instruction and data bandwidth. The claims are more specifically directed to methods and a processor for implementing conditional vector operations. Examples of these conditional vector operations include dividing an input vector into two or more output vectors based on a condition vector, combining one or more input vectors into a single output vector based on a condition vector, conditional vector switching, conditional vector combining, and conditional vector load balancing.

An example embodiment of the present invention is directed to a method of performing a distributed conditional vector input operation in a processor. *See, e.g.*, claim 10, FIGS. 5-6B, and the corresponding discussion at page 15, lines 5 *et seq*. The method

includes generating a plurality of electrical signals as a condition vector representative of whether individual arithmetic clusters in a plurality of arithmetic clusters are to receive data. *See, e.g.*, page 21, lines 4-6. Also included is distributing a plurality of electrical signals as an input vector having input vector elements to arithmetic clusters in the plurality of arithmetic clusters for which a corresponding portion of the condition vector is equal to a predetermined value, a length of the condition vector being greater than a length of the input vector. *See, e.g.*, page 21, lines 6-8. The method also includes using the arithmetic clusters to process the input vector elements distributed thereto and assembling the processed input vector elements to form an output vector having a length equal to that of the condition vector. *See, e.g.*, page 21, lines 8-15. Other variations of this embodiment are disclosed in claims 12 and 13. Claim 14 is another variation of the embodiment disclosed in claim 10 that further includes storage. *See, e.g.*, page 17, line 18 – page 19, line 14.

Another example embodiment of the present invention is directed to a method of performing a conditional vector switching operation in a processor. *See, e.g.*, claim 15, page 24, line 30 – page 27, line 17 and FIGs. 8A and 8B. The method includes receiving electrical signal representative of an input vector, the input vector having input vector elements and receiving electrical signals representative of a condition vector. The condition vector has condition vector elements and a number of the input vector elements are equal to a number of the condition vector elements, the input vector elements and the condition vector elements being in one-to-one correspondence with one another and each condition vector element being a result of evaluating a predetermined conditional expression using data corresponding to an input vector element. The method also includes generating electrical signals representative of an intermediate vector (*e.g.*, 260 of FIG. 3B). *See, e.g.*, page 26, line 31 – page 27, line 7. The intermediate vector has intermediate vector elements that contain input vector elements for which corresponding condition vector elements are equal to a predetermined value, a number of intermediate vector elements being equal to a number of condition vector elements in the condition vector that are equal to the predetermined value. The method further includes generating electrical signals representative of an output vector, the output vector having output vector elements in one-to-one correspondence with the intermediate vector elements, each output vector element is a result of a computation performed on each corresponding intermediate vector element. Other variations of this

embodiment are disclosed in claims 16 and 17, *e.g.*, claim 17 includes limitations directed to a second intermediate vector

Another example embodiment of the present invention is directed to a method of performing a distributed conditional vector load balancing operation in a processor. *See, e.g.*, claim 20, FIGs. 9A-B, and the corresponding discussion at page 30, *et seq*. The method includes distributing a plurality of electrical signals as an input vector, the input vector having input vector elements, to successive arithmetic clusters in a plurality of arithmetic clusters. Also included is, using the arithmetic clusters to process the input vector elements distributed thereto, each arithmetic cluster completing processing of one input vector element before another input vector element is distributed to the arithmetic cluster, the processing time of any input vector element being dependent on a value of the input vector element. *See, e.g.*, page 31, lines 26-30. Further, the processed input vector elements are assembled to form output vector elements of an output vector, each processed input vector element forming at least one output vector element. *See, e.g.*, page 31, line 31 – page 32, line 2. Another variation of this embodiment is disclosed in claim 21, *e.g.*, claim 21 includes limitations directed to operations performed for each input vector element.

Another example embodiment of the present invention is directed to a processor. *See, e.g.*, claim 24, FIG. 1 and the corresponding discussion at page 10, lines 6-14. The processor includes a first memory area to store input vector elements 16, a second memory area 14, a third memory area 18, and a logic circuit 20. The second memory area is used to store condition vector elements and has a same length as the first memory area. The third memory area is used to store output vector elements and comprises memory locations from plural clusters where the plural clusters generate the output vector elements. The third memory has a length equal to a number of condition vector elements that are equal to a predetermined value. The logic circuit transfers input vector elements from the first memory area to the third memory area when corresponding condition vector elements stored in the second memory area are equal to the predetermined value to generate the output vector elements in the plural clusters, wherein an output vector element is stored in a memory location of a different cluster of the plural clusters than that cluster which generated the output vector element.

Another example embodiment of the present invention is directed to a processor for performing conditional vector operations, including a conditional vector input operation and a conditional vector output operation. *See, e.g.,* claim 26, FIG. 1, FIG. 2 and the corresponding discussion at page 10, line 15-page 12, line 6. The processor includes three memory areas similar to those discussed above in connection with claim 24 (*e.g.*, 16, 14 and 18), a buffer (*e.g.*, 28), a plurality of processing elements (*e.g.*, 26), a switch (*e.g.*, 30), and a controller (*e.g.*, 20).

As required by 37 C.F.R. § 41.37(c)(1)(v), a concise explanation of the subject matter defined in the independent claims involved in the appeal is provided herein. Appellant notes that representative subject matter is identified for these claims; however, the abundance of supporting subject matter in the application prohibits identifying all textual and diagrammatic references to each claimed recitation. Appellant thus submits that other application subject matter, which supports the claims but is not specifically identified above, may be found elsewhere in the application. Appellant further notes that this summary does not provide an exhaustive or exclusive view of the present subject matter, and Appellant refers to the appended claims and their legal equivalents for a complete statement of the invention.

VI. Grounds of Rejection

A. Claims 15-17, 24 and 26 are rejected under 35 U.S.C. § 102(b) over Asai *et al.* (U.S. Patent No. 5,553,309).

B. Claims 10-14, 20 and 21 are rejected under 35 U.S.C. § 102(b) over Inagami *et al.* (U.S. Patent No. 4,881,168).

VII. Argument

With respect to both Section 102(b) grounds of rejection, the Examiner relies on an inconsistent and improper interpretation of the claims and ignores important claim terms. In order to satisfy the requirements for a *prima facie* Section 102(b) rejection of a claim, the Examiner must present evidence of correspondence between each of the claimed limitations and the cited reference. MPEP § 2131. The Examiner has failed to identify correspondence to several claim terms, and specifically the terms "computation" and "arithmetic cluster;" consequently, Appellant submits that each of the Section102(b) rejections is improper. Accordingly, Appellant requests that each of the rejections be reversed.

A. The Section 102(b) rejection of claims 15-17, 24 and 26 is improper because the Examiner fails to present correspondence between each of the claimed limitations and the cited '309 reference.

1. Claims 15-17

The Examiner fails to identify how the '309 reference teaches or suggests the claimed "computation" as used, for example, in the claim term "each output element is a result of a computation performed on each corresponding intermediate element." The Examiner's failure to cite any evidence in support thereof is consistent with the teaching of the '309 reference – which does not mention, teach or suggest whatsoever, the claimed "computation." Rather, the '309 reference teaches storing of an output element in a memory without involving any such computation.

As the Examiner's rejection does not explain why or how the '309 reference is being used to support the rejection, Appellant submits that the rejection either ignores the term "computation" altogether or is using a generalized definition of the term "computation" such that term is in effect, ignored. The term "computation" would be accepted by those having

ordinary skill in the art as referring to execution of an algorithm and/or a mathematical process – as this is the conventional definition of "computation" (*see, e.g.*, Evidence Appendix: Web definitions for **computation**).

Moreover, each of the claims is directed to a processor/computer implementation involving arithmetic logic for processing vector signals (data) in order to, *inter alia,* reduce computational complexity. Assuming that the Examiner's rejection uses an overly-generalized definition of "computation," when considering that the claims use this term in connection with arithmetic operations by a computer, such an overly-generalized definition removes any meaning for this term, as anything being done by a computer would be considered a "computation." The skilled artisan would not employ such an unsupported and out-of-context interpretation.

Further, the Examiner's rejection appears to be using a definition of "computation" that is contrary to use of the term in Appellant's specification and, therefore, contrary to MPEP 2173.02. In this regard, context for the "computation" has been clearly provided in the Specification at page 13, lines 4-6, where this term is explained as referring to execution of an arithmetic process, as follows: "The microcontroller 20 executes each of the kernels as an independent process using the arithmetic clusters 18 for performing ***computational*** operations." (emphasis added) This use of the term "computation" is consistent with the other related uses in the Specification, for example, in the Background section, the Specification uses this term and discusses it in the context of the problems and issues being addressed by the claimed invention. *See* page 2, lines 6-15 (especially lines 12-15), and page 2 line 16 through page 5.

Accordingly, the evidence of record contradicts any argument that might be offered by the Examiner in an attempt to support the rejection. As can be seen per the respective clauses at the bottom of each of the rejected claims, 15, 16 and 17, the claimed invention is limited to such "computation(s)." The '309 reference, however, fails to evidence any such correspondence. Therefore, Appellant submits that the rejection is improper and should be reversed.

2. Claim 24

The Examiner's rationale with respect to claim 24 relies upon an inconsistent interpretation of some of the claim limitations, specifically, limitations directed to the claimed "plural clusters" which are not taught by the '309 reference. At page 3 item 7 of the Office Action, the Examiner unnecessarily attempts to support the rejection as follows: "in Figure 6, component A(1) is a processing cluster that generates an output vector element A(1) in output vector register 2, component A(4) is a processing cluster that generates an output vector element A(4) in output vector register 2, and component A(5) is a processing cluster that generates an output vector element A(5) in output vector register 2, etc." Contrary to the Examiner's characterization, elements A(1), A(4), and A(5), *etc.* cannot be construed on one hand to be a "processing cluster" and on the other hand as data values per "an output vector element." Clearly, A(1), A(4), and A(5), etc. are values of an array A() with respective element indices 1, 4, and 5, *etc.* and they do not form a cluster. Accordingly, the Examiner fails to present correspondence at least to the claimed "plural clusters" and the rejection should be reversed.

3. Claim 26

The rejection of claim 26 is improper because a number of limitations have been ignored or misconstrued. At page 4 item 9 of the Office Action, the Examiner states: Applicant has not claimed a device that configures a switch "so that the switch is capable of transferring the input and output vector elements between any of the first plurality of entries of the buffer" and as such this limitation is not read into the claims. Appellant respectfully submits that the Examiner is mistaken because claim 26 reads in pertinent part (italics added):

> *a controller* to direct conditional vector input and output operations by controlling reading the input and the output vector elements from the buffer, by controlling receiving the input and the output vector elements into the buffer; by processing the condition vector elements, and by *configuring the switch so that the switch is capable of transferring the input and output vector elements between any of the first plurality of entries of the buffer*, any of the second plurality of entries of the buffer, and any of the plurality of processing elements.

Claim 26 clearly includes a switch capable of transferring the input and output vector elements between any of the first plurality of entries of the buffer; therefore, the Examiner's

characterization in item 9 on page 4 of the Office Action that "this argument is moot" is erroneous. Further, the '309 reference teaches a data selection circuit limited to receiving elements from an input vector register and sending elements to a compress circuit. The '309 reference does not teach nor suggest a switch capable of the claimed "transferring the input and output vector elements between any of the first plurality of entries of the buffer, any of the second plurality of entries of the buffer, and any of the plurality of processing elements." Without a presentation of correspondence to each of the claimed limitations, the rejection is improper and should be reversed.

B. The Section 102(b) rejection of claims 10-14, 20 and 21 is improper because the Examiner fails to present correspondence between each of the claimed limitations and the cited '168 reference.

1. Claims 10-14 and 20

The Examiner fails to present any correspondence from the '168 reference to the claimed "arithmetic cluster." The '168 reference does not teach or suggest any structure resembling these limitations. Appellant clearly makes apparent the subject matter relating to the claimed "arithmetic cluster" at, for example, page 11, lines 15-20, and FIG. 2 of the Specification. This disclosure is consistent with the requirements of MPEP § 608.01(o), which states that the meaning of every term used in the claims should be apparent from the descriptive portion of the specification and may be given a special meaning in the description. Appellant is not attempting to read in limitations of the claim language, as alleged by the Examiner in the Advisory Action, but rather is enforcing the proper interpretation of the claimed "arithmetic clusters." Without a presentation of correspondence to at least these claimed limitations, the Section 102(b) rejection cannot stand.

Appellant further submits that certain claim limitations have been ignored or misconstrued. More specifically, the '168 reference does not teach or suggest the claimed "length of the condition vector being greater than a length of the input vector." Appellant appreciates the Examiner's clarifications on page 4 items 10 and 11 of the Office Action alleging the length of the input buffer; however, Appellant respectfully disagrees with the Examiner's assessment that "the input buffer consists of ten elements, A0-A9." Figure 4,

element 21, of the '168 reference clearly shows sixteen elements A0-A15, the same number of elements as the mask vector 22. Thus, the '168 reference teaches that the mask vector and input buffer have equal lengths, which does not correspond to the claimed "length of the condition vector being greater than a length of the input vector." Without a presentation of correspondence to each of the claimed limitations, the rejection is improper and should be reversed.

2. Claim 21

The Examiner fails to identify teachings from the '168 reference that correspond to the claimed iteration because the iteration discussed in the '168 reference is not iterative processing, as claimed. On page 5 item 13 of the Office Action, the Examiner asserts that the '168 reference "have taught an iteration, see column 7, lines 8-14." At column 7, lines 8-14, the '168 reference teaches repeated steps to sequentially store elements of vector data, with each element stored exactly once. In contrast, claim 21 is directed to iterations that are directed to processing the same element multiple times. Thus, the repeated steps taught by the '168 reference do not correspond to the claimed iterative processing of the input vector. Without a presentation of correspondence to each of the claimed limitations, the rejection is improper and should be reversed.

VIII. Conclusion

In view of the above, Appellant submits that the rejections are improper, the claimed invention is patentable, and that the rejections of claims 10-17, 20, 21, 24 and 26 should be reversed. Appellant respectfully requests reversal of the rejections as applied to the appealed claims and allowance of the entire application.

Authority to charge the undersigned's deposit account was provided on the first page of this brief.

Respectfully submitted,

CRAWFORD MAUNU PLLC
1270 Northland Drive – Suite 390
St. Paul, MN 55120
(651) 686-6633

By: _____
Name: Robert J. Crawford
Reg. No. 32,122

APPENDIX OF APPEALED CLAIMS

10. A method of performing a distributed conditional vector input operation in a processor, the method comprising:

generating a plurality of electrical signals as a condition vector representative of whether individual arithmetic clusters in a plurality of arithmetic clusters are to receive data;

distributing a plurality of electrical signals as an input vector having input vector elements to arithmetic clusters in the plurality of arithmetic clusters for which a corresponding portion of the condition vector is equal to a predetermined value, a length of the condition vector being greater than a length of the input vector;

using the arithmetic clusters to process the input vector elements distributed thereto; and

assembling the processed input vector elements to form an output vector having a length equal to that of the condition vector.

11. A method according to claim 10 wherein the corresponding portion of the condition vector comprises corresponding condition vector elements, and wherein a certain plurality of arithmetic clusters receive input vector elements as a result of corresponding condition vector elements for the certain plurality of arithmetic clusters being equal to the predetermined value.

12. A method of performing a conditional vector input operation in a processor, the method comprising:

generating a plurality of electrical signals as a condition vector representative of whether individual arithmetic clusters in a plurality of arithmetic clusters are to receive data;

providing a plurality of electrical signals as an input vector having input vector elements to arithmetic clusters in the plurality of arithmetic clusters for which corresponding condition vector elements of the condition vector are

equal to a predetermined value, the number of clusters being greater than the number of input vector elements, the input vector elements being in one-to-one correspondence with corresponding condition vector elements of the condition vector that are equal to the predetermined value;

using the arithmetic clusters to process the input vector elements provided thereto; and

assembling the processed input vector elements to form an output vector.

13. A method of performing a conditional vector input operation in a processor, the method comprising:

generating a plurality of electrical signals as a condition vector representative of whether individual arithmetic clusters in a plurality of arithmetic clusters are to receive data;

providing a plurality of electrical signals as an input vector having input vector elements to arithmetic clusters in the plurality of arithmetic clusters for which corresponding condition vector elements of the condition vector are equal to a predetermined value, the number of clusters being greater than the number of input vector elements, the input vector having only input vector elements corresponding to condition vector elements of the condition vector that are equal to the predetermined value;

using the arithmetic clusters to process the input vector elements provided thereto; and

assembling the processed input vector elements to form an output vector.

14. A method of performing a distributed conditional vector input operation in a processor, the method comprising:

generating a plurality of electrical signals as a condition vector representative of whether individual arithmetic clusters in a plurality of arithmetic clusters are to receive data;

distributing a plurality of electrical signals from a storage area in at least one cluster as an input vector to arithmetic clusters in the plurality of

arithmetic clusters for which a corresponding portion of the condition vector is equal to a predetermined value, a length of the condition vector being greater than a length of the input vector;

using the arithmetic clusters to process input vector elements of the input vector distributed thereto;

assembling the processed input vector elements to form an output vector having a length equal to that of the condition vector; and

storing an output vector element of the output vector in a storage area of a cluster different from a cluster from which an input vector element was distributed.

15. A method of performing a conditional vector switching operation in a processor, the method comprising:

receiving electrical signals representative of an input vector, the input vector having input vector elements;

receiving electrical signals representative of a condition vector, the condition vector having condition vector elements, a number of the input vector elements being equal to a number of the condition vector elements, the input vector elements and the condition vector elements being in one-to-one correspondence with one another, and each condition vector element being a result of evaluating a predetermined conditional expression using data corresponding to an input vector element;

generating electrical signals representative of an intermediate vector, the intermediate vector having intermediate vector elements that contain input vector elements for which corresponding condition vector elements are equal to a predetermined value, a number of intermediate vector elements being equal to a number of condition vector elements in the condition vector that are equal to the predetermined value; and

generating electrical signals representative of an output vector, the output vector having output vector elements in one-to-one correspondence with the intermediate vector elements, each output vector element is a result of a

computation performed on each corresponding intermediate vector element.

16. A method of performing a conditional vector switching operation in a processor, the method comprising:

receiving electrical signals representative of an input vector of a first length, the input vector having input vector elements;

receiving electrical signals representative of a condition vector of a second length, the first and second lengths being equivalent, the condition vector having condition vector elements in one-to-one correspondence with the input vector elements, at least one of the condition vector elements being equal to a value other than a predetermined value, at least one of the condition vector elements being equal to the predetermined value;

generating electrical signals representative of an intermediate vector of a third length, the third length being less than the first length, the intermediate vector having intermediate vector elements in one-to-one correspondence with any condition vector elements being equivalent to the predetermined value, the intermediate vector elements respectively comprising any input vector elements corresponding to any condition vector elements that are equal to the predetermined value; and

performing a computation on each intermediate vector element to generate electrical signals representative of an output vector of a fourth length, the third and fourth lengths being equivalent.

17. A method of performing a conditional vector switching operation in a processor, the method comprising:

receiving electrical signals representative of an input vector, the input vector having input vector elements;

receiving electrical signals representative of a condition vector, the condition vector having condition vector elements, a number of the input vector elements being equal to a number of the condition vector elements, the input vector elements and the condition vector elements being in one-to-one correspondence with one another, and each condition vector element

being a result of evaluating a predetermined conditional expression using data corresponding to an input vector element;

generating electrical signals representative of a first intermediate vector, the first intermediate vector having first intermediate vector elements that contain input vector elements for which corresponding condition vector elements are equal to a first predetermined value, a number of first intermediate vector elements being equal to a first number of condition vector elements in the condition vector that are equal to the first predetermined value; and

generating electrical signals representative of a second intermediate vector, the second intermediate vector having second intermediate vector elements that contain input vector elements for which corresponding condition vector elements are equal to a second predetermined value, a number of second intermediate vector elements being equal to a second number of condition vector elements in the condition vector that are equal to the second predetermined value;

generating electrical signals representative of a first output vector, the first output vector having first output vector elements in one-to-one correspondence with the first intermediate vector elements, each first output vector element is a result of a first computation performed on each corresponding first intermediate vector element; and

generating electrical signals representative of a second output vector, the second output vector having second output vector elements in one-to-one correspondence with the second intermediate vector elements, each second output vector element is a result of a second computation performed on each corresponding second intermediate vector element.

20. A method of performing a distributed conditional vector load balancing operation in a processor, the method comprising:

distributing a plurality of electrical signals as an input vector, the input vector having input vector elements, to successive arithmetic clusters in a plurality of arithmetic clusters;

using the arithmetic clusters to process the input vector elements distributed thereto, each arithmetic cluster completing processing of one input vector element before another input vector element is distributed to the arithmetic cluster, the processing time of any input vector element being dependent on a value of the input vector element; and

assembling the processed input vector elements to form output vector elements of an output vector, each processed input vector element forming at least one output vector element.

21. A method of performing a distributed conditional vector load balancing operation in a processor, the method comprising:

distributing a plurality of electrical signals as an input vector, the input vector having input vector elements, to successive processing elements in a plurality of processing elements; and

generating a plurality of electrical signals representative of an output vector, the output vector having output vector elements; and

for any input vector element,

requesting the input vector element from the input vector;

reading the input vector element into one processing element of the plurality of processing elements;

processing the input vector element for at least one iteration;

generating at least one output vector element from the processed input vector element;

reading an immediately succeeding input vector element, if available, of the input vector elements into any successive processing element of the plurality of processing elements that has completed processing any input vector element;

requesting a successive input vector element, if available, of the input vector elements; and

reading the successive input vector element into the one processing element when the input vector element has been processed.

24. A processor comprising:

a first memory area to store input vector elements;

a second memory area to store condition vector elements, the second memory area having a same length as the first memory area;

a third memory area to store output vector elements, the third memory area comprising memory locations from plural clusters, the plural clusters generating the output vector elements, the third memory area having a length equal to a number of condition vector elements that are equal to a predetermined value; and

a logic circuit that transfers input vector elements from the first memory area to the third memory area when corresponding condition vector elements stored in the second memory area are equal to the predetermined value to generate the output vector elements in the plural clusters, wherein an output vector element is stored in a memory location of a different cluster of the plural clusters than that cluster which generated the output vector element.

26. A processor to perform conditional vector operations, including a conditional vector input operation and a conditional vector output operation, comprising:

a first memory area to store an input vector stream, the input vector stream having input vector elements;

a second memory area to store an output vector stream, the output vector stream having output vector elements;

a third memory area to store a condition vector stream, the condition vector stream having condition vector elements;

a buffer having a first plurality of entries and a second plurality of entries to store the input vector elements and the output vector elements;

a plurality of processing elements to process input vector elements into output vector elements;

a switch configured to transfer the input vector elements from the buffer to the processing elements and to transfer the output vector elements to the buffer, the buffer receiving the input vector elements from the first memory area, the

buffer receiving the output vector elements from the plurality of processing elements via the switch, the second memory area reading the output vector elements from the buffer, and the plurality of processing elements reading the input vector elements from the buffer via the switch in accordance with the condition vector elements; and

a controller to direct conditional vector input and output operations by controlling reading the input and the output vector elements from the buffer, by controlling receiving the input and the output vector elements into the buffer; by processing the condition vector elements, and by configuring the switch so that the switch is capable of transferring the input and output vector elements between any of the first plurality of entries of the buffer, any of the second plurality of entries of the buffer, and any of the plurality of processing elements.

EVIDENCE APPENDIX

1. Web definitions for computation (2 pages)
2. Results from web search for "what is computation" (2 pages)

RELATED PROCEEDINGS APPENDIX

None.

Web

Definitions of **computation** on the Web:

- calculation: the procedure of calculating; determining something by mathematical or logical methods
- calculation: problem solving that involves numbers or quantities
  www.cogsci.princeton.edu/cgi-bin/webwn

- Computation can be defined as finding a solution to a problem from given inputs by means of an algorithm. This is what the theory of computation, a subfield of computer science and mathematics, deals with. For thousands of years, computing was done with pen and paper, or chalk and slate, or mentally, sometimes with the aid of tables.
  en.wikipedia.org/wiki/Computation

- A computation denotes an executing program composed of one or more active threads. Each computation consists of a code closure which specifies its behaviour, an execution state which stores all control information related to the execution the computation and a data space that includes all the resources accessible by the computation.
  cui.unige.ch/~ecoopws/tpi/glos.html

- Has become an essential component of scientificl research. The great quantity and diversity of the data being generated by different technologies is daunting, and impossible to organize or oversee without computational assistance. Without effective and integrated databases to store and retrieve these data and advanced computational methods such as pattern recognition and other machine learning approaches to analyze and interpret them, the full implications of these data will not be realized.
  www.cyrdas.org/ci.glossary.html

- The manipulation of numbers or symbols according to fixed rules. Usually applied to the operations of an automatic electronic computer, but by extension to some processes performed by minds or brains. See also Cognitivism.
  www.informatics.susx.ac.uk/books/computers-and-thought/gloss/node1.html

- A computation is the application of a sequence of operations to a set of values to yield a value.
  burks.brighton.ac.uk/burks/pcinfo/progdocs/plbook/def.htm

- References: 37
  www.cs.unb.ca/profs/bremner/PolytopeBase/glossary/glossmain.html

- through Computer terminal
  www.onlineencyclopedia.org/site-map_2.html

Google Web Images Groups News Froogle Local[New!] more »

what is "computation" Search Advanced Search Preferences

Web Results **1 - 10** of about **21,600,000** for **what is "computation"**. (**0.18** seconds)

Web definitions for **computation**

calculation: the procedure of calculating; determining something by mathematical or logical methods

www.cogsci.princeton.edu/cgi-bin/webwn - Definition in context

Mathematics of **Computation**
Areas covered include numerical analysis, computational number theory and algebra, and related fields. Table of contents. Articles available to subscribers.
www.ams.org/mcom/ - Similar pages

Evolutionary **Computation** - The MIT Press
Featuring research on the theoretical and practical aspects of evolutionary computational systems.
mitpress.mit.edu/catalog/ item/default.asp?tid=25&ttype=4 - 17k - Cached - Similar pages

Neural **Computation** - The MIT Press
Multidisciplinary research on the twin scientific and engineering challenges of understanding the brain and building computers.
mitpress.mit.edu/catalog/ item/default.asp?tid=31&ttype=4 - 15k - May 7, 2005 - Cached - Similar pages

Neural **Computation**
Offers articles, titles, and reviews. Site has an archive going back to 1995.
neco.mitpress.org/ - Similar pages

Centre for Quantum **Computation**
Based at Oxford University. Includes extensive information.
www.qubit.org/ - 3k - Cached - Similar pages

VassarStats: Statistical **Computation** Web Site
Web site for statistical **computation**; probability; linear correlation and regression; chi-square; t-procedures; t-tests; analysis of variance; ANOVA; ...
faculty.vassar.edu/~lowry/VassarStats.html - 3k - Cached - Similar pages

Journal of Logic and **Computation**
Web site for Journal of Logic and **Computation**. ... For faster access to Journal of Logic and **Computation** from these countries use this URL: ...
logcom.oupjournals.org/ - Similar pages

Sunrise/Sunset **computation**
Sunrise/Sunset/Twilight and Moonrise/Moonset/Phase. This site has moved to the US Naval Observatory Astronomical Applications Dept.
tycho.usno.navy.mil/srss.html - 1k - Cached - Similar pages

Information and **Computation**
(Elsevier Science Direct) Contents and abstracts from vol.102 (1993).
www.sciencedirect.com/science/journal/08905401 - Similar pages

JSTOR: Mathematics of **Computation**
... Mathematics of **Computation** Cover Image. Mathematics of **Computation**. (continues Mathematical Tables and Other Aids to **Computation**). JSTOR Coverage: Vols. ...
www.jstor.org/journals/00255718.html - 6k - Cached - Similar pages

Gooooooooogle ▶
Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**

what is "computation" Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google